PCT WORLD INTELLECTUAL PROPERTY ORGANIZATION
International Bureau

INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(51) International Patent Classification 6 : H04M 2/00	A1	**(11) International Publication Number:** WO 97/32427 **(43) International Publication Date:** 4 September 1997 (04.09.97)

(21) International Application Number: PCT/US97/03329

(22) International Filing Date: 28 February 1997 (28.02.97)

(30) Priority Data:
08/609,699 1 March 1996 (01.03.96) US

(71) Applicant: NETPHONIC COMMUNICATIONS, INC. [US/US]; 1580 West El Camino Real, Mountain View, CA 94040 (US).

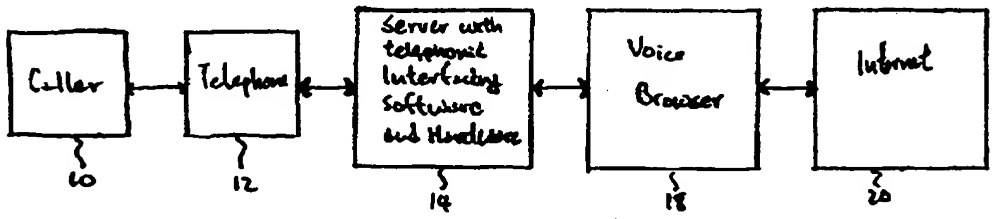
(72) Inventors: RHIE, Kyung, H.; 417 Becker Lane, Los Altos, CA 94022 (US). KWAN, Richard, J.; 41020 Amapola Court, Fremont, CA 94539 (US). OLSEN, Lee, E.; 289 Crows Nest Drive, Boulder Creek, CA 95006 (US). HAHN, John, S.; 905 Loyola Drive, Los Altos, CA 94024 (US).

(74) Agents: HAMRICK, Claude, A., S. et al.; Bronson, Bronson & McKinnon L.L.P., Suite 600, Ten Almaden Boulevard, San Jose, CA 95113 (US).

(81) Designated States: AU, CA, JP, KR, European patent (AT, BE, CH, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE).

Published
With international search report.

(54) Title: METHOD AND APPARATUS FOR TELEPHONICALLY ACCESSING AND NAVIGATING THE INTERNET



(57) Abstract

A method for accessing and browsing the internet (20) through the use of a telephone and the associated dtmf signals is disclosed. The preferred embodiment provides a system that converts the information content of a web page from text to speech (voice signals), signals the hyperlink selections of a web page in an audio manner (18), and allows selection of the hyperlinks through the use of dtmf signals generated from a telephone keypad. Upon receiving a dtmf signal corresponding to a hyperlink, the corresponding web page is fetched and again delivered to the user via one of the available delivery methods such as voice, fax-on-demand, electronic mail, or regular mail (26).

FOR THE PURPOSES OF INFORMATION ONLY

Codes used to identify States party to the PCT on the front pages of pamphlets publishing international applications under the PCT.

AM	Armenia	GB	United Kingdom	MW	Malawi
AT	Austria	GE	Georgia	MX	Mexico
AU	Australia	GN	Guinea	NE	Niger
BB	Barbados	GR	Greece	NL	Netherlands
BE	Belgium	HU	Hungary	NO	Norway
BF	Burkina Faso	IE	Ireland	NZ	New Zealand
BG	Bulgaria	IT	Italy	PL	Poland
BJ	Benin	JP	Japan	PT	Portugal
BR	Brazil	KE	Kenya	RO	Romania
BY	Belarus	KG	Kyrgystan	RU	Russian Federation
CA	Canada	KP	Democratic People's Republic of Korea	SD	Sudan
CF	Central African Republic			SE	Sweden
CG	Congo	KR	Republic of Korea	SG	Singapore
CH	Switzerland	KZ	Kazakhstan	SI	Slovenia
CI	Côte d'Ivoire	LI	Liechtenstein	SK	Slovakia
CM	Cameroon	LK	Sri Lanka	SN	Senegal
CN	China	LR	Liberia	SZ	Swaziland
CS	Czechoslovakia	LT	Lithuania	TD	Chad
CZ	Czech Republic	LU	Luxembourg	TG	Togo
DE	Germany	LV	Latvia	TJ	Tajikistan
DK	Denmark	MC	Monaco	TT	Trinidad and Tobago
EE	Estonia	MD	Republic of Moldova	UA	Ukraine
ES	Spain	MG	Madagascar	UG	Uganda
FI	Finland	ML	Mali	US	United States of America
FR	France	MN	Mongolia	UZ	Uzbekistan
GA	Gabon	MR	Mauritania	VN	Viet Nam

Specification

METHOD AND APPARATUS FOR TELEPHONICALLY ACCESSING AND NAVIGATING THE INTERNET

BACKGROUND OF THE INVENTION

Field of the Invention

The present invention relates generally to methods for retrieving information from an interconnected network and for accessing and delivering the retrieved information to a user, and, more particularly, a method for accessing and retrieving information from an interconnected networks such as the internet via a telephone in response to the user's request and for delivering the information via voice, fax-on-demand, e-mail, and other means to the user.

Description of the Prior Art

Under the conventional method of accessing information on an interconnected network such as the internet, the user is required to have a certain amount of computer software and hardware and is expected to have a certain level of computer expertise before the user can successfully access (or browse) a wide range of information now available on the internet. If the user does not have the necessary hardware and the appropriate software to direct the computer to establish a connection to the internet via a modem or a direct connection to the internet, the user would then have no other means available to him or her for accessing the internet.

Given the amount of information now readily available on the internet, having the ability to access the internet becomes a matter of convenience as well as a matter of having access to an invaluable information source.

Additionally, from a company or an organization point of view, it is advantageous to direct customers to a centralize information database and thereby necessitating the maintenance of only one database rather than multiple databases.

The software and hardware requirement for accessing the internet creates a barrier for most people to take advantages of this information source. Prior art systems overcome this problem by providing a telephone fax-on-demand system where a user uses a telephone to dial into a company's web page and directs the system to fax the web page back to the user. However, the manner in operating this type of system is tedious and time consuming. In order for the user to access a hyperlink on the web page, the first web page needs to be faxed back to the user with the hyperlinks numerically annotated for reference. The user then calls a second time (or wait for the first fax page to arrive on another line) to access subsequent web pages numerically using the now numbered hyperlinks.

It is thus clear that a better system is needed to access and browse the internet in an inexpensive and efficient manner.

SUMMARY OF THE INVENTION

It is therefore an object of the present invention to provide a method for accessing and browsing the internet through the use of a telephone.

It is another object of the present invention to provide a method for accessing and browsing the internet by converting the information content of a web page to voice format.

It is yet another object of the present invention to provide a method for signaling the user in an audio manner the hyperlink selections in a web page.

It is yet another object of the present invention to provide a method for accessing and browsing the internet where the information content of a web page may be provided to the user via voice format, fax-on-demand, e-mail, or regular mail.

Briefly, a method for accessing and browsing the internet through the use of a telephone and the associated DTMF signals is disclosed. The preferred embodiment of the present invention provides a system that converts the information content of a web page from text to speech (voice signals), signals the hyperlink selections of a web page in an audio manner, and allows selection of the hyperlinks through the use of DTMF signals as generated from a telephone keypad. Upon receiving a DTMF signal corresponding to a hyperlink, the corresponding web page is fetched and again delivered to the user via one of the available delivery methods.

An advantage of the present invention is that it provides a method for accessing and browsing the internet through the use of a telephone.

Another advantage of the present invention is that it provides a method for accessing and browsing the internet by converting the information content of a web page to voice format.

Yet another advantage of the present invention is that it provides a method for signaling the user in an audio manner the hyperlink selections in a web page.

Yet another advantage of the present invention is that it provides a method for accessing and browsing the internet where the information content of a web page may be provided to the user via voice format, fax-on-demand, e-mail, or regular mail.

These and other objects and advantages of the present invention will no doubt become obvious to those of ordinary skill in the art after having read the following illustrations and detailed description of the preferred embodiments.

IN THE DRAWINGS

Fig. 1 illustrates the components of the preferred embodiment of the present invention;

Fig. 2 shows the subsystems for the voice browser of the present invention;

Fig. 3 illustrates the subsystems of the HTree Generator/Web browser;

Fig. 4 shows the components of the Voice Data Management System; and

Fig. 5 illustrates the components of the fax data management system.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENTS

Referring to Fig. 1, the preferred embodiment of the present invention is operated by a caller 10 using a telephone 12 to dial into a server having telephonic interfacing software and hardware. The server provides audio directions to the caller and provides a number of options which may be selected by the corresponding DTMF tones generated through the use of a telephone keypad. By pressing a number on the keypad corresponding to the desired option, the caller selects one of the several available options. The server then receives the DTMF tone, converts the tone to a corresponding DTMF code, identifies the option corresponding to the code, and executes the action corresponding to the selected option. In this manner, the caller is able to direct the server to take certain available actions.

One of the available actions is to direct a voice browser 18 to navigate the internet 20. By selecting the voice browser, the caller is provided with an audio readback of a default web page where the available links to other web pages are read back to the user and are indicated by a special audio signal such as a short duration tone signal, a beep, a "bong" sound, etc.

For example, referring to Appendix D illustrating a listing of the code of the preferred embodiment as described below, instructions (starting from page 13 of Appendix D) are provided to direct a user to navigate through available documents and the hyperlinks therein. By repetitively selecting documents and the hyperlinks contained therein, the user can navigate the entire internet.

User Interface for Operating the Voice Web

To operate the preferred embodiment of the present invention, a touch-tone phone and the phone number to access a server are all that is required.

The voice browser provides a series of audio menus to guide the caller to retrieve documents or web pages from the internet. Several options are provided at each menu and the caller may make a selection by pressing the corresponding key(s) on the telephone. The opening menu may request a password before allowing the caller to access the system.

For inputs requiring specific spelling of the information being entered (e.g. e-mail address, name, street address, etc.), each character can be defined by pressing a two-key combination. The first key indicates the key where the letter appears, and the second key indicates the position it occurs on that key. For example, the letter "A" is defined as 21, "B" is defined as 22, "C" is defined as 23, "D" is defined as 31, etc. However, the letters "Q" and "Z" are not defined on a telephone keypad and they may be assigned by special two-key combinations. In the preferred embodiment, the letter "Q" is defined as 17 and the letter "Z" is defined as 19. Other special characters may be assigned as well. For example the "@" sign is defined as 12, the "_" character is defined as 18, the "." (period) is defined as 13, and a " " (space) is defined as 11.

The actual interface between the voice browser and the telephony interfacing hardware is expected to vary according to the implementation. There are two issues involved here. One issue involves interface control, namely how the software commands are accepted by the interface and how errors or exceptions are signaled. The other issue involves audio encoding -- how audio (e.g. voice) data is represented at the interface. For example, the Rockwell chip set utilizes a Hayes-compatible command set which is extended for fax and voice operations, and where audio data is encoded in the Adaptive Differential Pulse Coded Modulation (ADPCM) format. Under a Unix system, ioctl() commands may be used to manipulate the audio interface. There may be another protocol for ISDN lines as well.

Voice Web

Referring to Fig. 2, the voice browser is software driven and is composed of several cooperating subsystems. From the main engine 22, depending on the selection made or the scheduling algorithm, either the Call Menu Script Interpreter 24 of the document delivery subsystem 26 may be activated.

The Call Menu Script Interpreter (CMSI) guides a caller through a series of call menus and plays voice segments of retrieved web pages or documents to the caller. It responds to the caller's touch tone selection and drives the HTree Generator/Web Browser 28. The task is accomplished by first directing the web browser to retrieve the desired web page(s) from the internet. The HTree Generator then converts the retrieved web page into an HTree which is a data structure storing web page data in a particular format conducive for conversion into voice format. The CMSI then traverses the HTree, sending HTree segments to the Voice Data management Subsystem (Voice DMS) 38. The Voice DMS contains pre-recorded texts or text-strings. For the texts or text-strings not in the Voice DMS, a text-to-speech engine is provided to convert the text to speech. The information content of the web page is then delivered to caller in the voice format. Optionally, the web page can also be delivered to the caller in other available methods, or it can be marked as a document request job for later processing. If the caller is calling from a phone line connected to a fax machine, the web page can be immediately delivered to the caller via the fax machine.

The web browser 28 of the preferred embodiment is a general web browser modified to interface with the HTree Generator and to access the internet 20. The web page or documents that may be readily accessed by the caller are indexed by document-ID, title, or Universal Resource Locator (URL) and stored in a database.

The document delivery subsystem 26 responds to the several available methods for delivering the web page to the caller. These methods include delivery via fax, electronic mail, and regular postal mail. The document delivery subsystem 26 also may directly retrieve a web page as selected by a caller and deliver the information to the caller via one of the available delivery methods.

In the case delivery by fax, the documents may be directly retrieved by a Fax Data Management Subsystem (Fax DMS) where the web pages in the HTML format may be converted to the PostScript format and then from the PostScript format to TIFF image format.

In the case delivery by e-mail, documents may be directly retrieved by an E-Mail Data Management Subsystem where web pages in the HTML format may be converted to straight ASCII text or to the PostScript format.

Call Menu Script Interpreter

The Call Menu Script Interpreter (CMSI) guides the caller through a series of call menus via voice prompts and plays voice segments of a retrieved web page to the caller. The caller interacts with the CMSI by generating DTMF tones from the caller's touch-tone keypad. The DTMF tones are converted into ASCII text usually by the telephony interface hardware. In one embodiment, the conversion from DTMF tones to ASCII text is performed by a voice modem.

In order to carry out all of the tasks, the CMSI is implemented utilizing a set of software registers and instructions. A software register may be readable, writable, markable, or nav-writable where a nav-writable register allows the navigational mechanism of the browser to write to it. Registers which can be written by the DTMF-converted ASCII text are referred to as "caller-writable registers." These registers include the DocID, FaxNum, ExtNum, ExtName, Passwd, Userid, and Selection registers. Registers for which the ASCII text can be converted into voice are termed "caller-readable registers". These registers include the DocId, FaxNum, ExtNum, ExtName, and Title registers. Markable registers include the DocId and URL registers. The nav-writable registers include the DocId, URL, Title, MarkedNumDoc, MaxNumDoc, and MaxRepeatInput registers.

There is also an instruction set associated with the registers. There are four basic categories of instructions: Execution Control instruction, Caller Register instructions, Audio Output instructions, and Miscellaneous instructions. Some of the more basic instructions are the Play instruction which plays the content of a register, a prompt, or a web page; the Get instruction which gets the DTMF input into a register; the Mark instruction which marks a page or document according to a URL or DocId; the Goto instruction which unconditionally jumps to a different location in the script; and the Selection instruction which jumps if a given register matches a given DTMF string. Appendix A attached herein explains each of the registers and instructions. The preferred embodiment is not limited to the listed registers and instructions in Appendix A. New instructions and/or registers can be readily added to accommodate new features or functionalities.

As the caller navigates through the web, an URL list is kept. A maximum size limit is imposed on the URL list where a First-In-First-Out system is maintained. Additionally, each of the URL entry in the URL list may have an associated HTree. There is an HTree cache for all the HTree entries. For the HTree cache, if there is a limitation in size, the least recently used entries is deleted first.

HTree Generator / Web Browser

The HTree Generator/Web Browser (hereinafter "HTree Generator") is the Voice Browser's agent for accessing the World-Wide-Web. It is similar to a GUI-based web browser except that the user interface is replaced by an audio interface residing within the CMSI. The HTree Generator is driven by the CMSI. Referring to Fig. 3, the HTree Generator 28 is generally comprised of a generator interface component 50, an HTree Converter 52, and a web browser 54. The CMSI sends a URL 56 to the HTree Generator 28. The generator interface 50 receives the URL and sends the URL to the web browser 54 to retrieve the web page or document via one of the several available methods (e.g. FTP and HTTP). The

HTree Converter 52 converts the retrieved web page/document into an HTree, and the HTree is sent to the CMSI. Thus, regardless of the format of the retrieve document, it can be converted into HTree format for processing.

At its highest level, an HTree contains a series of HTree sentences. An HTree sentence can comprise several HTree segments which when strung together forms a complete sentence in the language of choice including English and Spanish. The top level structure, represented by the symbol _HTree, represents a given web page where the structure may include the URL of the page, its title, and it may include a number of linked lists. Appendix B attached herein provides a full explanation of the HTree structure.

Voice Data Management Subsystem

The Voice Data Management Subsystem (Voice DMS) provides audio segments to the CMSI for the given text segments. Referring to Fig. 4, the main interface between the Voice DMS and the CMSI is the voice interface 60. The voice interface 60 receives a text string from the CMSI and passes it to the Voice database 64 to retrieve the corresponding digitally encoded voice pattern. The voice interface 60 then returns the encoded voice pattern 66 to the CMSI. If the voice interface does not find the text string in the Voice database, it will send the text string to the Text-To-Speech subsystem 68 to generate the digitally encoded voice pattern for that text string. The generated voice pattern is passed to the voice interface to pass to the CMSI. Whenever the Text-To-Speech subsystem generates a voice pattern for a text string, the text string and the generated voice pattern is passed to the Voice database and stored for future reference.

Some of the voice interface's functions for driving the application program interface (API) include voiceopen, voiceget and voiceclose. The voiceopen function creates a voice handle for a given URL that enables the CMSI to keep track of the data during a session. The voiceget operation uses the voice handle to retrieve a voice pattern. The voiceclose function simply closes the interface associated with a voice handle.

Commonly used text strings may be recorded with human voice. A method of inputting text strings and associated human voice into the voice database involves the using of a voice teleprompter 70 and input interface hardware 72. Some of the capabilities that the teleprompter include Play, Start, Stop, and Record. The voice teleprompter 70 receives an HTree as input 74 and displays HTree segments on the teleprompter 70. A person whose voice is being recorded would read the text strings as displayed by the teleprompter 70 and use the input interface hardware 72 to input his or her voice. Once the recording is satisfactory, the teleprompter sends the recording along with the text string to the voice interface 60 for storage into the voice database 64.

Document Delivery Subsystem

The Document Delivery Subsystem provides a method for the caller to request a document or a web page to be delivered via one of the available delivery methods such as delivery via fax, via e-mail, or via regular postal service.

In delivering a document or web page via e-mail, the document or web page may be formatted for ASCII, a selected word processing program format, or another available format. In delivering a document or web page via regular postal mail, the system will ask for the caller's name and address if it is not already in a caller information database.

In delivering a document or web page via fax, once the TIFF files for all the requested documents have been retrieved or received, they files are concatenated into one file and queued for transmission. The fax can be sent to the caller right away if the caller has another phone line connected to a fax machine.

The document delivery subsystem operates like a queue where the queue is periodically check to see if there are any pending jobs. If there is a pending job, a scheduling file that contains the time for delivery, the method of delivery, and the file for delivery for each job is checked. If the transmission attempt is unsuccessful, the job will be rescheduled for transmission at a later time. There is a limit on the number of retries before the system administrator is notified. Appendix C attached herein provides the specific details for the document delivery subsystem.

Fax Data Management Subsystem

The Fax Data Management Subsystem (Fax DMS) interacts with the CMSI and Document Delivery Subsystem through an interface called the fax interface. Referring to Fig. 5, the fax interface 80 receives an URL 82 and returns a TIFF formatted file 84. The fax interface 80 searches the Fax database 86 for the corresponding TIFF file for the given URL. If a TIFF file is found, the fax interface retrieves the file and returns it to the requesting subsystem. If the requested file is not found in the Fax database, the fax interface sends the URL to a URL-To-TIFF convertor 88. The TIFF converter invokes a GUI browser 90 to retrieve the web page or document if it has not already been retrieved and uses the browser to convert the web page or document into a PostScript formatted file 92. The PostScript formatted file is then passed to a PostScript interpreter 94 which creates a TIFF file 96 from the PostScript file. The TIFF file is then passed back to the fax interface 80 and/or a fax database 86. Fax images can also be directly imported into the fax database through the use of a fax machine which creates an image capture file 100 for import into the fax database. The image capture file 100 ensures the quality of an image and may compare favorably against TIFF formatted images.

Although the present invention has been described in terms of the presently preferred embodiment, it is to be understood that such disclosure is not to be interpreted as limiting. Various alterations and modifications will no doubt become apparent to those skilled in the art after reading the above disclosure. Accordingly, it is intended that the appended claims be interpreted as covering all alterations and modifications as fall within the true spirit and scope of the invention.

What I claim is:

Appendix A: Call Menu Script Interpreter - Instruction Set

The Web-On-Call™ Call Menu Script Interpretor uses several registers.

A.1. Menu Registers

The table below describes each of the CMSI registers. The *Register Name* is the name of the register as defined in the code. The *Type* is the format of the field. The *creg-R/W/M* column denotes if the register is readable, writable, or markable by the Call Menu Script Interpreter. The *nav-W* column denotes whether the register can be written to by the navigation mechanisms of the browser.

Table 1: Menu Registers

Register Name	Type	creg-R/W/M	nav-W	Description
DocID	string	R/W/M	W	The document no. entered by the caller
FaxNum	string	R/W	-	The fax number of the caller
ExtNum	string	R/W	-	The extension number for the fax cover sheet
FaxName	string	R/W	-	The name for the fax cover sheet
Lang	string	-	-	The language selected for the prompts
Url	string	M	W	The current top of the URL stack
Title	string	R	W	The title of the page of the current URL
AltUrl	string		W	An optional alternate URL
AltTitle	string	R	W	The title of the alternate URL page
Passwd	string	W	-	The password of the caller
Userid	string	W	-	The userid of the caller
Selection	string	W	-	Input and conditional test variable
QuickSelection	string	W		
MarkedNumDoc	int	-	W	Number of documents marked for delivery
MaxNumDoc	int	-	W	Maximum number of deliverable documents
MaxRepeatInput	int	-	W	Maximum DTMF input retries

A.2. Instruction Set

The Call Menu Script language has four basic categories of instructions: Execution Control instructions, Caller Register instructions, Audio Output instructions, and Miscellaneous instructions.

In the description of the instruction set, the following variable definitions are used:

Table 2: Instruction Set Variable Definitions

creg-r	caller-readable register named "*creg-r*"
creg-w	caller-writable register named "*creg-w*"
creg-m	caller-markable register named "*creg-m*"
card	a cardinal number (non-negative number)
pfile	name of a prompt file
url	URL of a file
label	label name; target of a GOTO
ic	interpreter counter

A.2.1 Execution Control Instructions

LABEL *label*
Symbolic name for script location.

GOTO *label*
Continue execution at *label.*

EXIT
Terminate script execution.

NO_INPUT_GOTO *label*
If DTMF input is empty, continue execution *label.*

A.2.2 Caller Register Instructions

PLAY *creg-r*
Fetch voice segment for *creg-r* and play to telephony interface.

GET_PREVIOUS_PAGE

Pop the current top of the URL stack, so that the previous entry becomes the current value of the URL register.

COMMIT_KEY *key*

Cause the caller input string to be committed when *key* is received.

GET *creg-w card*
GET *creg-w creg-r*

In the first case, copy current DTMF string input to register *creg-w*. The string is up to length *card* characters, or is terminated by a commit key

In the second case, copy register *creg-r* to register *creg-w*. For example, GET Selection ExtNum would copy ExtNum to Selection, and allow later use of Selection in testing the value.

RESET *creg-w*

Reset value of *creg-w* to nil.

A.2.3. Audio Output Instructions

PLAY *creg-r*

See above description in "Caller Register Instructions."

PLAY PROMPT *pfile*

Fetch encoded voice for *pfile* and play to telephony interface.

PLAY PAGE

Fetch encoded voice for file corresponding to "url" and play to audio channel

PLAY DOC

Fetch voice segments for page corresponding to register "Url" and play to telephony interface. In the case of PLAY DOC, first convert value in register "DocID" to corresponding URL, and place in register "Url".

If a segment has an associated hypertext link, precede the voice segment by an audio "bong." If PLAY is interrupted, register "Url" gets the value of the hypertext link.

PLAY PAGE_CONTENT_ONLY

Fetch voice segments for page corresponding to register "Url" and play to telephony interface. Do not indicate hypertext links.

PLAY CATALOG

Fetch voice segment for file corresponding to catalog and play to telephony interface.

PLAY QUICK

Play the Quick Navigation list for the page corresponding to register "Url."

This command plays a list of links which are accessible from the current page. For each link, it generates a voice segment saying "for title, press number", where "title" is the title of the document corresponding to the URL on the link, and "number" is the two-digit number the caller can press.

A.2.4. Miscellaneous Instructions

CHECK_AVAIL_OK *label*

If all marked documents are currently available, then go to *label.*

CHECK_PASSWD_OK *label*

If "Passwd" is confirmed for "Userid", then go to "label", change "ie" accordingly.

CHECK_USER_ID_OK *label*

If "Userid" is a valid entry, then go to "label"; change "ie" accordingly.

COUNT_PASSWORD_INPUT_RETRY *label*

The password retry limit is set in the configuration file. If the number of attempts at password entry exceeds this password retry limit, then go to *label.*

COUNT_USER_ID_INPUT_RETRY *label*

The userid retry limit is set in the configuration file. If the number of attempts at userid entry exceeds this userid retry limit, then go to *label.*

POP_PAGE

Pop the current top of the URL stack, so that the previous entry becomes the current value of register "Url".

MARK *creg-m*

MARK CATALOG

Put the page indicated by register creg-m (or in the latter case, the document catalog) on the list of deliverable documents and increment "MarkedNumDoc".

MARK_DOCID

Put the page indicated by "Docid" on the list of deliverable documents and increment "MarkedNumDoc".

MARK_HTML_PAGE

Put the page indicated by the "Url" on the list of deliverable documents and increment "MarkedNumDoc".

MARK_QUICK_PAGE

Put the page indicated by the "Url" on the list of deliverable documents (This instruction is now obsolete).

MAX_NUM_DOC *card*

Set maximum number of documents deliverable to *card.*

(Note: *card* should be less than the system defined maximum number of documents. If MAX_NUM_DOC is not executed, the system will use the system-defined default.)

MAX_REPEAT_INPUT *card*

Set maximum number of repeated DTMF inputs to *card*.

If MAX_REPEAT_INPUT is not executed, the system will use the system-defined default.

PLAY_HTML_CONTENT_ONLY

Play the content of the page corresponding to "Url".

PUT_EXTENSION

Use "ExtNum" with the current list of deliverable documents.

PUT_NAME

Use "ExtName" with the current list of deliverable documents.

PUT *creg-w*

Write contents of *creg-w* onto the cover page of the deliverable documents. This is used for writing information such as extension number, extension name, etc.

QUICK_SELECTION *key label*
If "QuickSelection" equals *key*, then go to *label*.

RESET_QUICK_SELECTION
Reset the value of the "QuickSelection" register to Null pointer.

RESET_SELECTION
Reset the value of the "Selection" register to Null pointer.

SELECTION *key label*
If register "Selection" equals *key*, then go to *label*.

SEND *deliv-method*
Fetch requested documents in the format required by deliv-method, and immediately deliver them.

Currently, *deliv-method* can only be set to FAX. This fetches the TIFF images of the requested documents, and transmits them to the telephony interface.

SET_LANGUAGE *lang*
Set the language used for voice prompts to "lang".

QUEUE *deliv-method*
Enqueue the current list of requested documents for delivery by *deliv-method*.

The list of available *deliv-method* values is implementation defined. Typical values include: FAX, USMAIL, EMAIL.

TRANSFER_OPERATOR *phonenumber*
Transfer the caller to the phone number indicated by *phonenumber*.

VERIFY_DOCID_OK *label*
If the page corresponding to "DocID" is valid, then go to "label".

VERIFY_MAX_OK *label*
If "MarkedNumDoc" is less than "MaxNumDoc", then go to "label".

VERIFY_QUICK_SELECTION_OK *label*
If the value of "QuickSelection" is valid, then go to "label".

VERIFY *creg-w label*

VERIFY *nav-w label*

If the value in register *creg-w* is valid, go to *label*. The verification mechanism is implementation dependent. (This is useful in Userid/Passwd verification.)

For VERIFY MarkedNumDoc, confirm that "MarkedNumDoc" < "MaxNumDoc".

Appendix B: HTree Structure

The Web Browser receives information in the form of a URL from the Voice Navigator and generates an H-Tree based on the text from the corresponding web page.

The H-Tree contains at its highest level a series of H-Tree sentences. An H-Tree sentence can comprise several H-Tree segments, which, when strung together, will form a complete sentence in the English or Spanish language.

An H-Tree is a collection of data structures for communicating between a H-Tree web browser and a program which operates on voice segments.

The top level structure, _HTree, represents a given web page. It includes the URL of the page and its title. It also includes a number of linked lists. These handle:

- H-Tree sentences -- pseudo-sentences which are extracted from the web pages.
- HRef anchors -- hyperlinks which refer to off-page web content.
- Name anchors -- hyperlinks referring to points in the current web page.

An H-Tree sentence, represented by the structure _HTreeSent, is an ordered sequence of voice segments. When a program (e.g., the voice navigator) breaks its audio playback and is required to restart, it normally plays from the beginning of the current H-Tree sentence.

Voice segments are represented by H-Tree segments. Separate H-Tree segments are generated for each piece of text which has an anchor associated with it. (This corresponds to the common practice of underlined or highlighted text in a GUI-based web browser browser.)

H-Tree segments may also be defined to isolate relatively stable text from text which is frequently updated. For example, the text:

Stock NPCI is selling at $25 1/4

might be broken into the segments

Stock	(stable)
NPCI	(heavy update)
is selling at	(stable)
$25 1/4	(heavy update)

This would be accomplished by HTML tags at the appropriate points in an HTML file. This sequence of H-Tree segments would define an H-Tree sentence.

An H-Tree has a segment of text, and a potentially non-zero HRef number pointing into the HRef anchor list for the H-Tree sentence.

The HRef and Name anchor lists are both lists of H-Tree anchors. Each anchor has a URL and a title associated with the URL. The title may be either the text associted with the URL in the current web page, or the actual document title associated with the URL, as found in a URL-title table.

In its simplest form, an H-Tree sentence would correspond to a header, paragraph, or list item in an HTML file. Rules can be programmed into the H-Tree web browser to provide different, or finer divisions. Thus, instead of recognizing paragrahs as H-Tree sentences, it might be possible to recognize English sentences as H-Tree sentences. (Not that changing the language of the text might change the H-Tree sentence recognition rules.)

```
/* HTree anchors */
struct _HTreeAnc {                           /* anchors */
      struct _HTreeAnc * next;               /* siblings */
      struct _HTreeAnc * prev;
      char *url;                             /* universal res loc */
      char *title;                           /* title assoc with url */
};

/* HTree segments and sentences */
struct _HTreeSeg {                           /* text segment */
      struct _HTreeSeg * next;               /* siblings */
      struct _HTreeSeg * prev;
      char *text;                            /* actual text */
      shorthref_num;                         /* corresponding href or 0 */
};

struct HTreeSent {                           /* text "sentence" */
      struct _HTReeSent * next;              /* siblings */
      struct _HTReeSent * prev;
      struct _HTReeSeg * first;              /* children */
      struct _HTReeSeg * last;
};

/* The HTree structure */
struct _HTree {                              /* a document */
      char *        url;                     /* universal res loc */
      char *        title;                   /* document title */
      struct _HTreeSent *sent_first          /* first "sentence" */
      struct _HTreeSent (sent_last           /* last "sentence" */
      short         num_hrefs;               /* # of anchor hrefs */
      struct _HTreeAnc *href_first;          /* first anc href */
      struct _HTreeAnc *href_last;           /* last anc href */
      short         num_names;               /* # of anchor names */
      struct _HTreeAnc *name_first;          /* first anc name */
      struct _HTreeAnc *name_last;;          /* last anc name */
};
```

Appendix C: Document Request Job

A Document Request Job specifies documents to be delivered to a requesting party, and how delivery should take place. It is generated by the Call Menu Script Interpreter and operated on by the Document Delivery Sub-system.

C.1. Job Structure

The basic structure looks as follows:

```
struct DocRequest {
      int         req_type;        /*doc# or URL */
      char *      doc_ident
};

struct DocumentRequestJob {
      char *      deliv_method
      char *      address;
      int         Nrequests;
      struct DocRequest dreq[MAX_DOC_NUM];
}
```

Within the DocumentRequestJob structure,

deliv_method indicates the delivery method.
address is a delivery address known to the delivery method
Nrequests is the number of documents requested (used as a maximum index for dreq).
dreq is an array of DocRequest structure, each indicating how a caller identified the document (req_type) and the identification itself (doc_ident).

The table below shows the correspondence between deliv_method and address

Table 3:

deliv_method	address
fax two-call	fax number
fax one-call	empty
email	email address
postal service	Voice recording or prestored postal address
express courier	Voice recording or prestored postal address

For each DocRequest, if the requested document is identified in `doc_ident` by DocID, then the Document Delivery sub-system will convert the document number into the corresponding URL for subsequent retrieval of the document in the format requested by the caller.

C.2 Example

This example shows a request of documents to be delivered by fax two-call.

```
struct DocumentRequestJob request = {
      "fax two-call",
      "19001234567",
      3,
      { docnum,  "101" },
      { URL,     "http://localhost/genlinfo/index.html" },
      { docnum,  "105" },
      { URL,     "http://localhost/genlinfo/hrpolicy.html" }
};
```

APPENDIX D

Psudo-code of the described preferred embodiment.

README - gives some minor help in deciphering the pseudocode.
main.cc - initializes Web-On-Call, then sends it on its way.
WOC.cc - top-level routine; this is actually the "main engine" mentioned in chapter 2 of functional design spec.
CMSI.cc - Call Menu Script Interpretor. This is an stripped down model which illustrates some of the instructions of the interpretor.
VoiceDMS.cc - Voice Data Management Subsystem; pretty solid.
FaxDMS.cc - Fax Data Management Subsystem; pretty solid.
menu.small - sample script used by Call Menu Script Interpretor.

```
----------
X-Sun-Data-Type: readme-file
X-Sun-Data-Description: readme-file
X-Sun-Data-Name: README
X-Sun-Charset: us-ascii
X-Sun-Content-Lines: 21


=================================
README for Web-On-Call pseudocode
=================================

revision:  @(#)README 1.1 96/02/10 NetPhonic

This code provides a conceptual model of what is or should be
going on in the Web-On-Call product.  No code from the real
product is reproduced here.  However, it is crafted as compilable
C++ code.  This helps in:
      * clarifying the essential objects of the product

      * ensuring that the represented algorithms have a semblance
      of reality to them

Pure pseudo-code, with no executable content is bracketed as
conditional source, e.g.,

    #ifdef pseudocode
 unexecutable pseudo-code goes here
    #endif // pseudocode

Sometimes, there will be an "else" clause with "stub" code.  This
allows the overall model to compile and run, in spite of the lack
of
complete code.  For example,
    #ifdef pseudocode
 unexecutable pseudo-code goes here
    #else // above pseudocode; below stubcode
 executable stub code
    #endif // pseudocode
```

Statements of the form

```
    cout << "something" << endl;
```

are output statements to allow tracing through execution, and may be
semi-informative in terms of what is going on.

```
----------
X-Sun-Data-Type: default
X-Sun-Data-Description: default
X-Sun-Data-Name: main.cc
X-Sun-Charset: us-ascii
X-Sun-Content-Lines: 21

//
// Pseudo-code for main program
//

#pragma ident "@(#)main.cc 1.1 96/02/12 NetPhonic"

#include <iostream.h>
#include "WOC.hh"
#include "Delivery.hh"

WOC * woc = 0;

int
main (int argc, char** argv)
{
    woc = new WOC;
    woc->modeloop();

    return 0;
}

----------
X-Sun-Data-Type: default
X-Sun-Data-Description: default
X-Sun-Data-Name: WOC.cc
X-Sun-Charset: us-ascii
X-Sun-Content-Lines: 73

//
// Pseudo-code for Web-on-Call top-level code
//
```

```
 #pragma ident "@(#)WOC.cc 1.1 96/02/12 NetPhonic"

 #include <iostream.h>
 #include "CMSI.hh"
 #include "WOC.hh"

 WOC::WOC()
 {
     cout << "WOC initializing..." << endl;

     interpretor = new CMSI;
     delivery = new Delivery;
 }

 void
 WOC::modeloop()
 {
     int timeout = 120;
     static int dummyruns = 0; // dummy counter for debugging
     const  int dummymax = 2; // dummy max for debugging
     Boolean got_caller;

     interpretor->read_script();

     while ( 1 ) {  // loop forever
 if (delivery->check_jobs(fax))
      delivery->deliver_a_job(fax);
 got_caller = poll_for_call (timeout);
 if (got_caller)
      interpretor->run();

#ifdef pseudocode
 ignore this
#else // above pseudocode; below stubcode
 if (++dummyruns>dummymax) break; // not part of design
#endif // pseudocode
     }
}

Boolean
WOC::poll_for_call(int tmo)
{
    Boolean got_call;
    static Boolean bdummy = False;

    cout << "WOC:  wait for call or timeout after "
 << tmo << " seconds." << endl;
#ifdef pseudocode
    set timer
    wait for events (timeout in tmo sec | incoming call)
    if (event == timeout)
 got_call = False;
```

Appendix D - Page 3

```
    else if (event == incoming call)
  got_call = True;
 #else // above pseudocode; below stubcode
    bdummy = bdummy?False:True; // dummy exercise code
    got_call = bdummy;
    cout << (bdummy?"got a caller":"no caller") << endl;
 #endif // pseudocode

    return got_call;
 }

CMSI *
WOC::get_interpretor()
{
    return interpretor;  // not needed in pseudocode
    // this is only here to satisfy "private" attribute in C++
}
----------
X-Sun-Data-Type: default
X-Sun-Data-Description: default
X-Sun-Data-Name: CMSI.cc
X-Sun-Charset: us-ascii
X-Sun-Content-Lines: 178

//
// Pseudo-code for Call Menu Script Interpretor
//

#pragma ident "@(#)CMSI.cc 1.2 96/02/16 NetPhonic"

#include <string.h>
#include <stdlib.h>
#include <iostream.h>
#include "Basic.hh"
#include "Charstr.hh"
#include "CMSI.hh"

extern CMSI_instruction script_input[];
extern void sample_script();
extern Charstr homepage;

ostream&
operator<< (ostream& s, const CMSIReg& reg)
{
    char* rstr;
    switch (reg) {
      case RegScratch: rstr = "RegScratch"; break;
      case RegVersion: rstr = "RegVersion"; break;
      case RegDocID: rstr = "RegDocID"; break;
      case RegFaxNum: rstr = "RegFaxNum"; break;
      case RegExtName: rstr = "RegExtName"; break;
      case RegLang: rstr = "RegLang"; break;
```

```
        case RegUrl: rstr = "RegUrl"; break;
        case RegPrompt: rstr = "RegPrompt"; break;
        case RegSelection:rstr = "RegSelection"; break;
        case RegTitle: rstr = "RegTitle"; break;
        case RegAltUrl: rstr = "RegAltUrl"; break;
        case RegAltTitle: rstr = "RegAltTitle"; break;
        case RegPasswd: rstr = "RegPasswd"; break;
        case RegUserid: rstr = "RegUserid"; break;
        default:  rstr = ""; break;
    }
    return s << rstr;
}

CMSI::CMSI()
{
    cout << "Call Menu Script Interp initialization begun." <<
endl;

    cout << "reset registers and clear memory" << endl;
    memset (code, 0, CMSI_codespace_sz*sizeof(CMSI_instruction));

    cout << "assign initial URL";
    Url = homepage;

    cout << "Call Menu Script Interp initialization completed."
<< endl;
}

void
CMSI::read_script()
{
    cout << "read/translate script into CMSI code" << endl;
    sample_script();
    cout << "read/translate script completed" << endl;
}

static Boolean
inbounds (int lo, int vx, int hi)
{
    int iret = ((lo <= vx) && (vx <= hi)) ? True : False;
    return (iret==1)?True:False;
}

void
CMSI::run()
{
    // this runs thru call script, generates doc request job

    cout << "Call Menu Script Interp execution begun." << endl;

    ic = 0;
    while ((control != quit) &&
 (inbounds(0,ic,CMSI_codespace_sz-1))) {
  cout << "Interpret ic=" << ic
       << " opcode=" << code[ic].opcode << endl;
  ic_next = ic+1;
  control = instruction (code[ic]);
  ic = ic_next;
      }

     cout << "Call Menu Script Interp execution completed." <<
endl;

}

ExecCntl
CMSI::instruction (const CMSI_instruction& inst)
{
    ExecCntl next_execcntl = fetch_and_interpret;

    inst_valid = False;  // assume inst requires validation
    switch (inst.opcode) {

      // Control operations
       case GoTo:
 ic_next = resolve_label (inst.opnd1.loc);
 break;
       case Exit:
 cout << "Exit" << endl;
 next_execcntl = quit;
 break;

       case Play:
 if (inst.opnd1.tag==OpRegister) {
     if (inst.opnd1.reg==RegPrompt) {
          if (inst.opnd2.tag==OpInteger) {
      DTMFlimit = inst.opnd2.ival;
      if (inst.opnd3.tag==OpCharstr) {
   Prompt = inst.opnd3.cstr;
   inst_valid = True;
      }
      // else instruction is no op
  } else if (inst.opnd2.tag==OpCharstr) {
      DTMFlimit = 0;
      Prompt = inst.opnd2.cstr;
      inst_valid = True;
  }
  if (inst_valid == True) {
      cout << "play prompt " << DTMFlimit << " "
   << Prompt << endl;
  }
     }
} else if (inst.opnd1.tag==OpPage) {
    inst_valid = True;
```

```
        cout << "page url=<" << Url << ">" << endl;
    }
    break;

        case Reset:
    if (inst.opnd1.tag==OpRegister) {
        inst_valid = True;
        cout << "reset " << inst.opnd1.reg << endl;
    }
    break;

        case Get:
    if (inst.opnd1.tag==OpRegister) {
        inst_valid = True;
        cout << "get " << inst.opnd1.reg << endl;
    }
    break;

        case Put:
    if (inst.opnd1.tag==OpRegister) {
        inst_valid = True;
        cout << "put " << inst.opnd1.reg << endl;
    }
    break;

    // Miscellaneous operations
        case Select:
          if ((inst.opnd1.tag==OpCharstr) &&
        (inst.opnd2.tag==OpCharstr)) {
        cout << "chkpt2" << endl;
        int loc = resolve_label (inst.opnd2.cstr);
        inst_valid = True;
        cout << "select " << endl;
        cout << inst.opnd1.cstr << endl << inst.opnd2.cstr << endl;
        cout << "select " << inst.opnd1.cstr << " "
    << inst.opnd2.cstr << endl;
        if (Selection == Charstr(inst.opnd1.cstr)) {
    ic_next = loc;
    cout << "selection taken" << endl;
        }
    }
    break;

        default:
          cout << "CMSI::instruction encounters unknown opcode" <<
endl;
 // unknown opcode
 break;
     }
     return next_execcntl;
}
----------
```

Appendix D - Page 7

```
X-Sun-Data-Type: default
X-Sun-Data-Description: default
X-Sun-Data-Name: VoiceDMS.cc
X-Sun-Charset: us-ascii
X-Sun-Content-Lines: 131

//
// Pseudo-code for Voice Data Management Subsystem
//

#pragma ident "@(#)VoiceDMS.cc 1.2 96/02/10 NetPhonic"

#include <string.h>
#include <stdlib.h>
#include <iostream.h>
#include "VoiceDMS.hh"

//
-----------------------------------------------------------------
// Text-to-Speech engine
//

TTSEngine::TTSEngine()
{
    cout << "connect to text-to-speech engine" << endl;
}

TTSEngine::~TTSEngine()
{
    cout << "disconnect from text-to-speech engine" << endl;
}

AudioSeg
TTSEngine::synthesize(Charstr& text)
{
    AudioSeg* as;
    cout << "TTSEngine synthesizes text: " << text << endl;
#ifdef pseudocode
    send text to TTS engine
    as = result of TTS synthesis
#else // above pseudocode; below stubcode
    as = new AudioSeg;// represents synthesized segment
#endif // pseudocode

    return *as;
}

//
-----------------------------------------------------------------
// VoiceDB
//
```

```
 VoiceDB::VoiceDB (Charstr& requested_url)
 {
      url = requested_url;
      cout << "VoiceDB opens for " << url << endl;
 }

 VoiceDB::~VoiceDB ()
 {
      cout << "VoiceDB closes for " << url << endl;
 }

 AudioSeg
 VoiceDB::fetch (Charstr& text)
 {
      AudioSeg aseg;

      cout << "VoiceDB fetches audio for " << text << endl;
#ifdef pseudocode
      aseg = fetch_record (key=text)
      if (segment found)
  aseg = found segment
      else
  aseg = zero length AudioSeg
#else // above pseudocode; below stubcode
  aseg.length = 0;
#endif // pseudocode

      return aseg;
 }

void
VoiceDB::store (Charstr& text, AudioSeg& audio)
{
      cout << "VoiceDB stores audio for " << text << endl;
#ifdef pseudocode
      if (record for key=text exists)
  delete existing record
      store record into VoiceDB with "text" as key, and "audio" as
data
#endif // pseudocode
}


//
--------------------------------------------------------------------
// VoiceFace
//

VoiceFace::VoiceFace()
{
      cout << "VoiceFace opens session." << endl;
#ifdef pseudocode
```

Appendix D - Page 9

```
     connect to text-to-speech engine
#else // above pseudocode; below stubcode
     tts_connection = new TTSEngine;
#endif // pseudocode
     current_url = 0;
}

VoiceFace::~VoiceFace()
{
     cout << "VoiceFace closes session" << endl;
#ifdef pseudocode
     disconnect from text-to-speech engine
#else // above pseudocode; below stubcode
     delete tts_connection;
#endif // pseudocode
}

AudioSeg
VoiceFace::fetch(Charstr& requested_url, Charstr& text)
{
     AudioSeg aseg;

     cout << "VoiceFace requests text at url" << endl;
     if (requested_url != current_url) {
 delete voicedb;  // close current DB
 voicedb = new VoiceDB(requested_url); // open new one
     }
     aseg = voicedb->fetch (text);
     if (aseg.length == 0) {
 AudioSeg newseg;
 // text segment not recorded yet; must synthesize
 newseg = tts_connection->synthesize (text);
 voicedb->store (text, newseg);
 aseg = newseg;
     }
     return aseg;
}
----------
X-Sun-Data-Type: default
X-Sun-Data-Description: default
X-Sun-Data-Name: FaxDMS.cc
X-Sun-Charset: us-ascii
X-Sun-Content-Lines: 138

//
// Pseudo-code for Fax Data Management Subsystem
//

#pragma ident "@(#)FaxDMS.cc 1.1 96/02/10 NetPhonic"

#include <string.h>
#include <stdlib.h>
```

```
#include <iostream.h>
#include "FaxDMS.hh"

//
------------------------------------------------------------------
// Webpage-to-Fax engine
// This may be a single engine serving several WOC browsers
// or multiple engines.  Single engine is used in this example
// code.
// Mutual exclusion lock code has been left out for clarity.
Besides,
// lock code placement can vary with how tuning is done.

WTFEngine::WTFEngine()
{
    cout << "connect to webpage-to-fax engine" << endl;
}

WTFEngine::~WTFEngine()
{
    cout << "disconnect from webpage-to-fax engine" << endl;
}

PostScriptImage
WTFEngine::web_to_ps(Charstr& url)
{
    PostScriptImage* ps;
    cout << "WTFEngine converting webpage to PS" << endl;
#ifdef pseudocode
    send url to WTF engine
    ps = PostScript after conversion from webpage
#else // above pseudocode; below stubcode
    ps = new PostScriptImage;// represents synthesized segment
#endif // pseudocode

    return *ps;
}

TiffImage
WTFEngine::ps_to_tiff(PostScriptImage& ps)
{
    TiffImage* tiff;
    cout << "WTFEngine converting PS to Tiff" << endl;
#ifdef pseudocode
    send PostScriptImage to WTF engine
    tiff = TIFF image after conversion from PostScript
#else // above pseudocode; below stubcode
    tiff = new TiffImage;// represents synthesized segment
#endif // pseudocode

    return *tiff;
}
```

```
//
----------------------------------------------------------------------------
// FaxDB
//

FaxDB::FaxDB (Charstr& requested_url)
{
     url = requested_url;
     cout << "FaxDB opens for " << url << endl;
}

FaxDB::~FaxDB ()
{
     cout << "FaxDB closes for " << url << endl;
}

TiffImage
FaxDB::fetch ()
{
     TiffImage tiff;

     cout << "FaxDB fetches TIFF for " << url << endl;
#ifdef pseudocode
     if (TiffImage exists)
 tiff = existing TiffImage
     else
 tiff = zero length TiffImage
#endif // pseudocode

     return tiff;
}

void
FaxDB::store (Charstr& newurl, TiffImage& newtiff)
{
     cout << "FaxDB stores TIFF for " << newurl << endl;
#ifdef pseudocode
     if (TIFF Image for newurl already exists)
 delete existing TIFF Image
     store newtiff into FaxDB
#endif // pseudocode
}


//
----------------------------------------------------------------------------
// FaxFace
//

FaxFace::FaxFace()
{
     cout << "FaxFace opens session." << endl;
}

FaxFace::~FaxFace()
{
    cout << "FaxFace closes session." << endl;
}

TiffImage
FaxFace::fetch (Charstr& requested_url)
{
    TiffImage tiff;

    cout << "FaxFace requests TiffImage" << endl;
    if (requested_url != current_url) {
 delete faxdb;  // close current DB
 faxdb = new FaxDB (requested_url); // open new one
    }
    tiff = faxdb->fetch();
    if (tiff.length == 0) {
 TiffImage newtiff;
 PostScriptImage newps;
 // image doesn't exist; must generate
 newps = wtf_connection->web_to_ps (requested_url);
 newtiff = wtf_connection->ps_to_tiff (newps);
 faxdb->store (requested_url, newtiff);
 tiff = newtiff;
    }
    return tiff;
}
----------
X-Sun-Data-Type: default
X-Sun-Data-Description: default
X-Sun-Data-Name: menu.small
X-Sun-Charset: us-ascii
X-Sun-Content-Lines: 700

#################################################################
##
# menu.small -- intended for script engine debugging
# %W% %E%
#################################################################
##

COMMIT_KEY #
COMMIT_KEY *


LABEL L_MAIN_MENU
 RESET_SELECTION
 PLAY_FILE 1 main_menu
# ..............................................................
# "To listen to general information about this site, press 1
```

Appendix D - Page 13

```
 #   To navigate quickly to a document of your interest, press 2
 #   If you already know the document number, press 3
 #   To obtain a document index, press 4
 # ..............................................................
  GET_SELECTION
    SELECTION 1 L_GENERAL_INFO
    SELECTION 2 L_QUICK_NAVIGATION
    SELECTION 3 L_DOCID
    SELECTION 8 L_MAIN_MENU
    SELECTION * L_MAIN_MENU
    PLAY_VERSION
    SELECTION ? L_INVALID_MAIN_MENU
    GOTO L_MAIN_NO_INPUT

LABEL L_INVALID_MAIN_MENU
 RESET_SELECTION
 PLAY_FILE 0 invalid_main_menu
# ...........................................................
# "I'm sorry.  That selection is not valid.  Please try again.
# ............................................................
 GOTO L_MAIN_MENU

LABEL L_MAIN_NO_INPUT
 RESET_SELECTION
 PLAY_FILE 0 main_menu_no_input
# ...........................................................
# "I'm sorry.  I did not hear any selection.  Please try again."
# ............................................................
 GOTO L_MAIN_MENU

##################################################################
#

LABEL L_DOCID
 RESET_DOCID
 PLAY_FILE 20 enter_docid
# .............................................................
# "Please enter a document number followed by the pound key.
#   If you do not know the document number, return to the
#   main menu for a document index."
# .............................................................
 GET_DOCID
 NO_INPUT_GOTO L_DOCID_NO_INPUT
 SET_NEW_PAGE
 PUT_NEW_PAGE
 GOTO L_VERIFY_DOCID

LABEL L_DOCID_NO_INPUT
 RESET_DOCID
 PLAY_FILE 0 docid_no_input
# .............................................................
# "I'm sorry.  I did not hear you enter the document number.
#  Plase try again."
# .............................................................
 GOTO L_DOCID

LABEL L_VERIFY_DOCID
 VERIFY_DOCID_OK L_VALID_DOCID
 GET_PREVIOUS_PAGE
 GOTO L_INVALID_DOCID

LABEL L_INVALID_DOCID
 PLAY_FILE 0 docid
# ..........................................................
# "I'm sorry.  The document number you entered:
# ..........................................................
 PLAY_DOCID
 RESET_SELECTION
 PLAY_FILE 1 docid_not_valid
# ..........................................................
# "is not valid.
#  To enter another document number, press 1.
#  To go back to the Top-level menu, press T or 8.
# ..........................................................
LABEL L_INVALID_DOCID_GOTO
 GET_SELECTION
  SELECTION 1 L_DOCID
  SELECTION 8 L_MAIN_MENU
  SELECTION ? L_INVALID_INVALID_DOCID
  GOTO L_INVALID_DOCID_NO_INPUT

LABEL L_INVALID_INVALID_DOCID
 RESET_SELECTION
 PLAY_FILE 0 invalid_invalid_docid
# ..........................................................
# "I'm sorry.  The selection you have made and
#  the document number you entered are not
#  valid.  Please try again.
#  To enter another document number, press 1.
#  To go back to the Top-level menu, press T or 8.
# ..........................................................
 GOTO L_INVALID_DOCID_GOTO

LABEL L_INVALID_DOCID_NO_INPUT
 RESET_SELECTION
 PLAY_FILE 0 invalid_docid_no_input
# ..........................................................
# "I'm sorry.  I did not hear you make any selection.
#  Please try again.
#  To enter another document number, press 1.
#  To go back to the Top-level menu, press T or 8.
# ..........................................................
 GOTO L_INVALID_DOCID_GOTO
 LABEL L_VALID_DOCID
  PLAY_FILE 0 you_entered
 # ..................................................
 # "You entered"
 # ..................................................
  PLAY_DOCID
  PLAY_FILE 0 titled
 # ..........................................................
 # "titled"
 # ..........................................................
  PLAY_URL_TITLE
 LABEL L_VALID_DOCID_GOTO
  RESET_SELECTION
  PLAY_FILE 1 docid_confirm
 # ..........................................................
 # "To confirm this document number, press 1
 #  To enter another document number, press 2
 #  To go back to the Top-level menu, press T or 8
 # ..........................................................
  GET_SELECTION
   SELECTION 1 L_DOCID_MENU
   SELECTION 2 L_UNDO_DOCID
   SELECTION 8 L_MAIN_MENU
   SELECTION ? L_INVALID_VALID_DOCID
   GOTO L_VALID_DOCID_NO_INPUT

 LABEL L_UNDO_DOCID
  GET_PREVIOUS_PAGE
  GOTO L_DOCID

 LABEL L_INVALID_VALID_DOCID
  RESET_SELECTION
  PLAY_FILE 0 invalid_main_menu
  GOTO L_VALID_DOCID_GOTO

 LABEL L_VALID_DOCID_NO_INPUT
  RESET_SELECTION
  PLAY_FILE 0 main_menu_no_input
  GOTO L_VALID_DOCID_GOTO

 LABEL L_DOCID_MENU
  RESET_SELECTION
  PLAY_FILE 1 docid_menu
 # ..........................................................
 # "To listen to the document, press 1
 #  To obtain a copy of this document, press 2
 #  If you finished selecting, press 3"
 # ..........................................................
  GET_SELECTION
   SELECTION 1 L_READ_DOC
   SELECTION 2 L_MARK_DOCID
   SELECTION 3 L_DELIVERY
```

```
    SELECTION 8 L_MAIN_MENU
    SELECTION * L_DOCID_MENU
    SELECTION ? L_INVALID_DOCID_MENU
    GOTO L_DOCID_MENU_NO_INPUT

LABEL L_INVALID_DOCID_MENU
  RESET_SELECTION
  PLAY_FILE 0 invalid_main_menu
  GOTO L_DOCID_MENU

LABEL L_DOCID_MENU_NO_INPUT
  RESET_SELECTION
  PLAY_FILE 0 main_menu_no_input
  GOTO L_DOCID_MENU

LABEL L_READ_DOC
  PLAY_FILE 0 read_doc
# ...........................................................
# "The system will now read the document content.
#  At any time during this narration, you can request a menu
#  by pressing the star key."
# ...........................................................
  READ_DOC
  GOTO L_DOCID_MENU

LABEL L_MARK_DOCID
  MARK_DOCID
  PLAY_FILE 0 receive_copy
# ......................................................
# "You will receive a copy of this document titled:"
# ......................................................
  PLAY_URL_TITLE
  GOTO L_DOCID_MORE

LABEL L_DOCID_MORE
  RESET_SELECTION
  PLAY_FILE 1 docid_more
# .......................................................
# "If you finished selecting, press 1
#  To select more documents, press 2"
# .......................................................
  GET_SELECTION
    SELECTION 1 L_DELIVERY
    SELECTION 2 L_DOCID
    SELECTION 8 L_MAIN_MENU
    SELECTION * L_DOCID_MORE
    SELECTION ? L_INVALID_DOCID_MORE
    GOTO L_DOCID_MORE_NO_INPUT

LABEL L_INVALID_DOCID_MORE
  RESET_SELECTION
  PLAY_FILE 0 invalid_main_menu
  GOTO L_DOCID_MORE

 LABEL L_DOCID_MORE_NO_INPUT
  RESET_SELECTION
  PLAY_FILE 0 main_menu_no_input
  GOTO L_DOCID_MORE

#####################################################################
##

LABEL L_QUICK_NAVIGATION
 PLAY_FILE 0 quick_navigation
# .........................................................
# "Starting with the home document, the system will now
#  read the available hyperlinks to related documents.
#  To select a hyperlink, press the pound key as soon as
#  you hear it,
#  OR enter the corresponding selection number.
#  When you arrive at a document of your interest,
#  you can retrieve the content of a document by pressing
#  the star key.
#  You are about to hear a list of hyperlinked navigation
#  choices available from within a document titled:"
# .........................................................
 PLAY_URL_TITLE
 RESET_QUICK_SELECTION
LABEL L_QUICK_NAVIGATION_GOTO
 PLAY_QUICK
 GET_QUICK_SELECTION
  QUICK_SELECTION *  L_QUICK_MENU
  QUICK_SELECTION #  L_VALID_QUICK_DOC
  QUICK_SELECTION ?  L_QUICK_DOC_SELECTED
  GOTO L_QUICK_NAVIGATION_NO_INPUT

LABEL L_QUICK_NAVIGATION_NO_INPUT
 RESET_QUICK_SELECTION
 PLAY_FILE 2 quick_navigation_no_input
# .........................................................
# "I'm sorry.  I did not hear you make any selection.
#  Please try again.
#  You are about to hear a list of hyperlinked navigation
#  choices available from within a document titled:"
# .........................................................
 PLAY_URL_TITLE
 GOTO L_QUICK_NAVIGATION_GOTO

LABEL L_QUICK_DOC_SELECTED
 VERIFY_QUICK_SELECTION_OK L_VALID_QUICK_DOC
 GOTO L_INVALID_QUICK_DOC

LABEL L_VALID_QUICK_DOC
 SET_NEW_PAGE
```

```
  PUT_NEW_PAGE
  PLAY_FILE 0 select_document_title
 # ............................................................
 # "you selected a document titled"
 # ............................................................
  PLAY_URL_TITLE
  RESET_SELECTION
  PLAY_FILE 1 quick_confirm
 # ............................................................
 # "To confirm this selection, press 1
 #  To enter another selection, press 2"
 #  To go back to the Top-level menu, press T or 8.
 # ............................................................
 GET_SELECTION
   SELECTION 1 L_PLAY_QUICK_GOTO
   SELECTION 2 L_QUICK_PREVIOUS_PAGE
   SELECTION 8 L_MAIN_MENU
   SELECTION * L_VALID_QUICK_DOC
   SELECTION ? L_INVALID_VALID_QUICK_DOC
   GOTO L_VALID_QUICK_DOC_NO_INPUT

LABEL L_PLAY_QUICK_GOTO
 PLAY_FILE 0 play_quick_goto
# ............................................................
# "You are about to hear a list of hyperlinked navigation
#  choices available from within a document titled:"
# ............................................................
 RESET_QUICK_SELECTION
 PLAY_URL_TITLE
 GOTO L_QUICK_NAVIGATION_GOTO

LABEL L_INVALID_VALID_QUICK_DOC
 RESET_SELECTION
 PLAY_FILE 0 invalid_main_menu
 GOTO L_VALID_QUICK_DOC

LABEL L_VALID_QUICK_DOC_NO_INPUT
 RESET_SELECTION
 PLAY_FILE 0 main_menu_no_input
 GOTO L_VALID_QUICK_DOC


LABEL L_INVALID_QUICK_DOC
 RESET_SELECTION
 PLAY_FILE 1 invalid_quick_doc
# ............................................................
# "I'm sorry.  Your selection is not valid."
#  To enter another selection, press 1.
#  To go back to the Top-level menu, press T or 8."
# ............................................................
LABEL L_INVALID_QUICK_DOC_GOTO
 GET_SELECTION
```

```
    SELECTION 1 L_PLAY_QUICK_GOTO
    SELECTION 8 L_MAIN_MENU
    SELECTION ? L_INVALID_QUICK_DOC
    GOTO L_INVALID_QUICK_DOC_NO_INPUT

  LABEL L_INVALID_QUICK_DOC_NO_INPUT
   RESET_SELECTION
   PLAY_FILE 0 invalid_quick_doc_no_input
  # .....................................................
  # "I'm sorry.  I did not hear you make any selection.
  #  Please try again.
  #  To enter another selection, press 1.
  #  To go back to the Top-level menu, press T or 8."
  # .....................................................
   GOTO L_INVALID_QUICK_DOC_GOTO

  LABEL L_QUICK_MENU
   RESET_SELECTION
   PLAY_FILE 1 quick_navigation_menu
  # ...........................................................
  # "To listen to the document content, press 1
  #  To obtain a copy of the document, press 2
  #  If you finished navigating, press 3
  #  To go back to the previous document, press 4
  #  To resume listing navigation choices, press 5
  #  To list all navigation choices from the beginning, press 6
  # ...........................................................
   GET_SELECTION
    SELECTION 1 L_PLAY_QUICK_DOC
    SELECTION 2 L_MARK_QUICK_DOC
    SELECTION 3 L_DELIVERY
    SELECTION 4 L_QUICK_PREVIOUS_PAGE
    SELECTION 5 L_QUICK_CONTINUE
    SELECTION 6 L_PLAY_QUICK_GOTO
    SELECTION 8 L_MAIN_MENU
    SELECTION * L_QUICK_MENU
    SELECTION ? L_INVALID_QUICK_MENU
    GOTO L_QUICK_MENU_NO_INPUT

 LABEL L_PLAY_QUICK_DOC
  PLAY_FILE 0 read_doc
  PLAY_HTML_CONTENT_ONLY
  PUT_NEW_PAGE
  GOTO L_QUICK_MENU

 LABEL L_MARK_QUICK_DOC
  MARK_QUICK_PAGE
  PLAY_FILE 0 receive_copy
  PLAY_URL_TITLE
  GOTO L_QUICK_MORE

 LABEL L_QUICK_MORE
 RESET_SELECTION
 PLAY_FILE 1 docid_more
# ...................................................................
# "If you finished selecting, press 1
#  To select more documents, press 2"
# ...................................................................
 GET_SELECTION
  SELECTION 1 L_DELIVERY
  SELECTION 2 L_PLAY_QUICK_GOTO
  SELECTION 8 L_MAIN_MENU
  SELECTION * L_QUICK_MORE
  SELECTION ? L_INVALID_QUICK_MORE
  GOTO L_QUICK_MORE_NO_INPUT

LABEL L_INVALID_QUICK_MORE
 RESET_SELECTION
 PLAY_FILE 0 invalid_main_menu
 GOTO L_QUICK_MORE

LABEL L_QUICK_MORE_NO_INPUT
 RESET_SELECTION
 PLAY_FILE 0 main_menu_no_input
 GOTO L_QUICK_MORE

LABEL L_QUICK_PREVIOUS_PAGE
 GET_PREVIOUS_PAGE
 SET_NEW_PAGE
 GOTO L_QUICK_NAVIGATION_GOTO

LABEL L_QUICK_CONTINUE
 RESET_QUICK_SELECTION
 GOTO L_QUICK_NAVIGATION_GOTO

LABEL L_INVALID_QUICK_MENU
 RESET_SELECTION
 PLAY_FILE 0 invalid_main_menu
 GOTO L_QUICK_MENU

LABEL L_QUICK_MENU_NO_INPUT
 RESET_SELECTION
 PLAY_FILE 0 main_menu_no_input
 GOTO L_QUICK_MENU

####################################################################
##

LABEL L_GENERAL_INFO
 RESET_SELECTION
 PLAY_FILE 1 general_info
# ...................................................................
# "You will now hear the content of the home document
#  and be able to switch to other related documents
```

```
 #  through hyperlinks.
 #  To proceed with this menu choice, press 1
 #  To go back to the Top-level menu, press T or 8
 # ..................................................
  GET_SELECTION
   SELECTION 1 L_GENERAL_PROCEED
   SELECTION 8 L_MAIN_MENU
   SELECTION * L_GENERAL_INFO
   SELECTION ? L_INVALID_GENERAL_INFO
   GOTO L_GENERAL_INFO_NO_INPUT

LABEL L_INVALID_GENERAL_INFO
 RESET_SELECTION
 PLAY_FILE 0 invalid_main_menu
 GOTO L_GENERAL_INFO

LABEL L_GENERAL_INFO_NO_INPUT
 RESET_SELECTION
 PLAY_FILE 0 main_menu_no_input
 GOTO L_GENERAL_INFO

LABEL L_GENERAL_PROCEED
 PLAY_FILE 0 general_instruction
# ..................................................
# "While listening, an audio signal will sound when a
#  hyperlink choice is available.
#  To navigate to any hyperlink, press the pound key
#  immediately after the signal and before the next one.
#  To obtain a copy of any document you will be listening to
#  or to find out available menu choices, press the star key."
# ..................................................
LABEL L_GENERAL_GOTO
 RESET_SEGMENT
 PLAY_FILE 0 ready_to_listen
# ..................................................
# "You are about to listen to a document titled"
# ..................................................
 PLAY_URL_TITLE

LABEL L_GENERAL_CONTINUE
 RESET_SELECTION
 PLAY_HTML_FILE
 GET_SELECTION
  SELECTION 8 L_MAIN_MENU
  SELECTION * L_GENERAL_INFO_MENU
  SELECTION # L_NEW_URL_GENERAL
  SELECTION ? L_INVALID_GENERAL_CONTINUE
  GOTO L_GENERAL_CONTINUE_NO_INPUT

LABEL L_INVALID_GENERAL_CONTINUE
 RESET_SELECTION
 PLAY_FILE 1 invalid_general_continue
 # ..........................................................
 # "I'm sorry.  Your selection is not valid.
 #  To find out available menu choices, press the star key.
 #  To listen to the document again, press 1.
 # ..........................................................
  GET_SELECTION
   SELECTION 1 L_GENERAL_PROCEED
   SELECTION 8 L_MAIN_MENU
   SELECTION * L_GENERAL_INFO_MENU
   SELECTION ? L_INVALID_GENERAL_CONTINUE
   GOTO L_GENERAL_CONTINUE_NO_INPUT

LABEL L_GENERAL_CONTINUE_NO_INPUT
 RESET_SELECTION
 PLAY_FILE 0 main_menu_no_input
 GOTO L_GENERAL_PROCEED

LABEL L_NEW_URL_GENERAL
 SET_NEW_PAGE
 PUT_NEW_PAGE
 GOTO L_GENERAL_GOTO

LABEL L_GENERAL_INFO_MENU
 RESET_SELECTION
 PLAY_FILE 1 general_menu
# ..........................................................
# "To listen to this document from the beginning, press 1
#  To obtain a copy of this document, press 2
#  If you have finished, press 3
#  To go back to the previous document, press 4
#  To resume listening where you have stopped, press 5
# ..........................................................
 GET_SELECTION
  SELECTION 1 L_GENERAL_REPLAY
  SELECTION 2 L_MARK_DOC_PAGE
  SELECTION 3 L_DELIVERY
  SELECTION 4 L_GENERAL_PREVIOUS_PAGE
  SELECTION 5 L_GENERAL_CONTINUE
  SELECTION 8 L_MAIN_MENU
  SELECTION * L_GENERAL_INFO_MENU
  SELECTION ? L_INVALID_GENERAL_INFO_MENU
  GOTO L_GENERAL_INFO_MENU_NO_INPUT

LABEL L_INVALID_GENERAL_INFO_MENU
 RESET_SELECTION
 PLAY_FILE 0 invalid_main_menu
 GOTO L_GENERAL_INFO_MENU

LABEL L_GENERAL_INFO_MENU_NO_INPUT
 RESET_SELECTION
 PLAY_FILE 0 main_menu_no_input
 GOTO L_GENERAL_INFO_MENU
```

```
  LABEL L_GENERAL_REPLAY
   SET_TO_ACTIVE_URL
   GOTO L_GENERAL_GOTO

  LABEL L_GENERAL_PREVIOUS_PAGE
   GET_PREVIOUS_PAGE
   SET_NEW_PAGE
   GOTO L_GENERAL_GOTO

  LABEL L_MARK_DOC_PAGE
   MARK_HTML_PAGE
   PLAY_FILE 0 receive_copy
   PLAY_URL_TITLE
   GOTO L_GENERAL_MORE

  LABEL L_GENERAL_MORE
   RESET_SELECTION
   PLAY_FILE 1 general_more
 # ......................................................
 # "If you have finished, press 1
 #  To select more documents, press 2"
 # ......................................................
   GET_SELECTION
    SELECTION 1 L_DELIVERY
    SELECTION 2 L_GENERAL_GOTO
    SELECTION 8 L_MAIN_MENU
    SELECTION * L_GENERAL_MORE
    SELECTION ? L_INVALID_GENERAL_MORE
   GOTO L_GENERAL_MORE_NO_INPUT

 LABEL L_INVALID_GENERAL_MORE
  RESET_SELECTION
  PLAY_FILE 0 invalid_main_menu
  GOTO L_GENERAL_MORE

 LABEL L_GENERAL_MORE_NO_INPUT
  RESET_SELECTION
  PLAY_FILE 0 main_menu_no_input
  GOTO L_GENERAL_MORE

##############################################################

LABEL L_DELIVERY
 PLAY_FILE 24 fax_number
# ......................................................
# "Please enter your fax number followed by the pound key.
# ......................................................
 GET_FAX_NUMBER
 NO_INPUT_GOTO L_TWO_CALL_NO_INPUT
 GOTO L_FAX_NUMBER_CONFIRM

LABEL L_TWO_CALL_NO_INPUT
```

```
 PLAY_FILE 0 two_call_no_input
# ..........................................................
# "I'm sorry. I did not hear you enter any fax number.
#  Please try again.
# ..........................................................
 GOTO L_TWO_CALL

LABEL L_FAX_NUMBER_CONFIRM
 PLAY_FILE 0 you_entered
 PLAY_FAX_NUMBER
 RESET_SELECTION
 PLAY_FILE 1 confirm_number
# ..........................................................
# "To confirm this number, press 1
#  To reenter the number, press 2
#  To select another delivery method instead, (such as
#  fax or eMail), press 3."
# ..........................................................
 GET_SELECTION
  SELECTION 1 L_EXTENSION_INPUT
  SELECTION 2 L_TWO_CALL_RESET
  SELECTION 3 L_DELIVERY
  SELECTION 8 L_MAIN_MENU
  SELECTION * L_FAX_NUMBER_CONFIRM
  SELECTION ? L_INVALID_FAX_NUMBER_CONFIRM
  GOTO L_FAX_NUMBER_CONFIRM_NO_INPUT

LABEL L_INVALID_FAX_NUMBER_CONFIRM
 RESET_SELECTION
 PLAY_FILE 0 invalid_main_menu
 GOTO L_FAX_NUMBER_CONFIRM

LABEL L_FAX_NUMBER_CONFIRM_NO_INPUT
 RESET_SELECTION
 PLAY_FILE 0 main_menu_no_input
 GOTO L_FAX_NUMBER_CONFIRM

LABEL L_EXTENSION_INPUT
 RESET_SELECTION
 PLAY_FILE 1 extension_selection
# ..........................................................
# "Your requested fax can be identified with
#  your telephone extension on the fax cover page.
#  To enter an extension, press 1,
# ..........................................................
 GET_SELECTION
  SELECTION 1 L_EXTENSION_NUMBER
  SELECTION 8 L_MAIN_MENU
  SELECTION * L_EXTENSION_INPUT
  SELECTION ? L_INVALID_EXTENSION_INPUT
  GOTO L_EXTENSION_INPUT_NO_INPUT
```

```
LABEL L_INVALID_EXTENSION_INPUT
 RESET_SELECTION
 PLAY_FILE 0 invalid_main_menu
 GOTO L_EXTENSION_INPUT

LABEL L_EXTENSION_INPUT_NO_INPUT
 RESET_SELECTION
 PLAY_FILE 0 main_menu_no_input
 GOTO L_EXTENSION_INPUT

LABEL L_EXTENSION_NUMBER_RESET
 RESET_EXTENSION_NUMBER
LABEL L_EXTENSION_NUMBER
 PLAY_FILE 64 enter_extension
# ..........................................................
# "Please enter your telephone extension followed by the
#  pound key"
# ..........................................................
 GET_EXTENSION_NUMBER
 NO_INPUT_GOTO L_EXTENSION_NUMBER_NO_INPUT
 GOTO L_EXTENSION_GOTO

LABEL L_EXTENSION_NUMBER_NO_INPUT
 PLAY_FILE 0 extension_number_no_input
# ..........................................................
# "I'm sorry. I did not hear you enter any telephone extension
#  Please try again."
# ..........................................................
 GOTO L_EXTENSION_NUMBER

LABEL L_EXTENSION_GOTO
 PLAY_FILE 0 you_entered
 PLAY_EXTENSION_NUMBER
 RESET_SELECTION
 PLAY_FILE 1 confirm_number
 GET_SELECTION
  SELECTION 1 L_EXTENSION_AND_DELIVERY
  SELECTION 2 L_EXTENSION_NUMBER_RESET
  SELECTION 8 L_MAIN_MENU
  SELECTION * L_EXTENSION_GOTO
  SELECTION ? L_INVALID_EXTENSION_GOTO
  GOTO L_EXTENSION_GOTO_NO_INPUT

LABEL L_INVALID_EXTENSION_GOTO
 RESET_SELECTION
 PLAY_FILE 0 invalid_main_menu
 GOTO L_EXTENSION_GOTO

LABEL L_EXTENSION_GOTO_NO_INPUT
 RESET_SELECTION
 PLAY_FILE 0 main_menu_no_input
 GOTO L_EXTENSION_GOTO
```

```
LABEL L_EXTENSION_AND_DELIVERY
 PUT_EXTENSION
 PLAY_FILE 0 two_call_fax_good_bye
# ..................................................
# "Your requested documents will be faxed to you
#  momentarily. There may be some delay in sending the
#  fax if there are many other requests ahead of you.
#  Good-bye !
# ..................................................
 TWO_CALL_SEND_FAX
 EXIT
```

CLAIMS

1. A method for telephonically accessing and retrieving information from an interconnected network of computers where each of said computers has one or more data files each accessible via an unique address, comprising the steps of:

receiving telephonically a request for accessing and retrieving information from an interconnected network of computers;

providing information telephonically in audio format of one or more addresses corresponding to one or more data files distributed in said interconnected network of computers, wherein each of said data files is in a first format;

receiving a signal corresponding to a particular address in said one or more addresses;

fetching the data file corresponding to said particular address;

converting said fetched data file in said first format to a second format; and

delivering said fetched data file in said second format.

2. A method as recited in claim 1 wherein the fetched data file includes one or more hyperlink texts having corresponding addresses to data files and one or more text segments each comprising a number of words, said converting step includes the substeps of:

reading said one or more hyperlink texts and said one or more text segments;

upon reading a hyperlink text, providing an audio signal identifying the occurrence of a hyperlink text; and

upon reading a text segment, placing the words in the text segment in a data structure and providing an audio voice representing one or more of the words in said data structure, said data structure allowing playing back of said words in said second format.

3. A method as recited in claim 1 wherein said first format is the Hyper Text Mark-up Language.

4. A method as recited in claim 1 wherein said second format is the speech voice signal.

5. A method as recited in claim 1 wherein said second format is the fax data format.

6. A method as recited in claim 1 wherein said second format is an electronic document format.

7. A method as recited in claim 1 wherein said second format is a paper-based document format.

8. A method as recited in claim 4 wherein said delivering step is carried out via a communication line.

9. A method as recited in claim 5 wherein said delivering step is carried out via a communication line.

10. A method as recited in claim 6 wherein said delivering step is carried out via said interconnected network of computers.

11. A method as recited in claim 7 wherein said delivering step is carried out via postal mail.

12. A method as recited in claim 4 wherein said voice signal is the English speech.

13. A method as recited in claim 4 wherein said voice signal is the Spanish speech.

14. A method as recited in claim 1 wherein said information in audio format includes the title for each of the addresses and an audio signal signifying that an address is available for accessing a data file corresponding to the address.

15. A method as recited in claim 14 wherein said audio signal is a single recorded signal.

16. A method as recited in claim 14 wherein said audio signal is a voice message.

17. A method as recited in claim 14 wherein said audio signal is a numeric voice message.

18. A method as recited in claim 2 wherein said provided voice signal is retrieved from an voice signal database.

19. A method as recited in claim 2 wherein said provided audio voice signal is generated from the words read from the datafile.20. A system for telephonically accessing and retrieving information from an interconnected network of computer where each of said computer has one or more data files each accessible via an unique address, comprising:

means for receiving telephonically a request for accessing and retrieving information from an interconnected network of computers;

means for providing information telephonically in audio format of one or more addresses corresponding to one or more data files distributed in said interconnected network of computers, wherein each of said data files is in a first format;

means for receiving a signal corresponding to a particular address in said one or more addresses;

means for fetching the data file corresponding to said particular address to create a fetched data file;

means for converting said fetched data file in said first format to a second format; and

means for delivering said fetched data file in said second format.

21. A system as recited in claim 20 wherein the fetched data file includes one or more hyperlink texts having corresponding addresses to data files and one or more text segments each comprising a number of words, said converting means includes:

means for reading said one or more hyperlink texts and said one or more text segments;

means for providing an audio signal identifying the occurrence of a hyperlink text upon reading a hyperlink text; and

means for placing the words in the text segment in a data structure and providing an audio voice representing one or more of the words in said data structure upon reading a text segment, said data structure allowing playing back of said words in said second format.

22. A system as recited in claim 20 wherein said first format is the Hyper Text Mark-up Language.

23. A system as recited in claim 20 wherein said second format is the speech voice signal.

24. A system as recited in claim 20 wherein said second format is the fax data format.

25. A system as recited in claim 20 wherein said second format is an electronic document format.

26. A system as recited in claim 20 wherein said second format is a paper-based document format.

27. A system as recited in claim 23 wherein said delivering means uses a communication line.

28. A system as recited in claim 24 wherein said delivering means uses a communication line.

29. A system as recited in claim 25 wherein said delivering means uses said interconnected network of computers.

30. A system as recited in claim 26 wherein said delivering means uses postal mail.

31. A system as recited in claim 23 wherein said voice signal is the English speech.

32. A system as recited in claim 23 wherein said voice signal is the Spanish speech.

33. A system as recited in claim 20 wherein said information in audio format includes the title for each of the addresses and an audio signal signifying that an address is available for accessing a data file corresponding to the address.

34. A system as recited in claim 33 wherein said audio signal is a single recorded signal.

35. A system as recited in claim 33 wherein said audio signal is a voice message.

36. A system as recited in claim 33 wherein said audio signal is a numeric voice message.

37. A system as recited in claim 21 wherein said provided voice signal is retrieved from an voice signal database.

38. A system as recited in claim 21 wherein said provided audio voice signal is generated from the words read from the datafile.

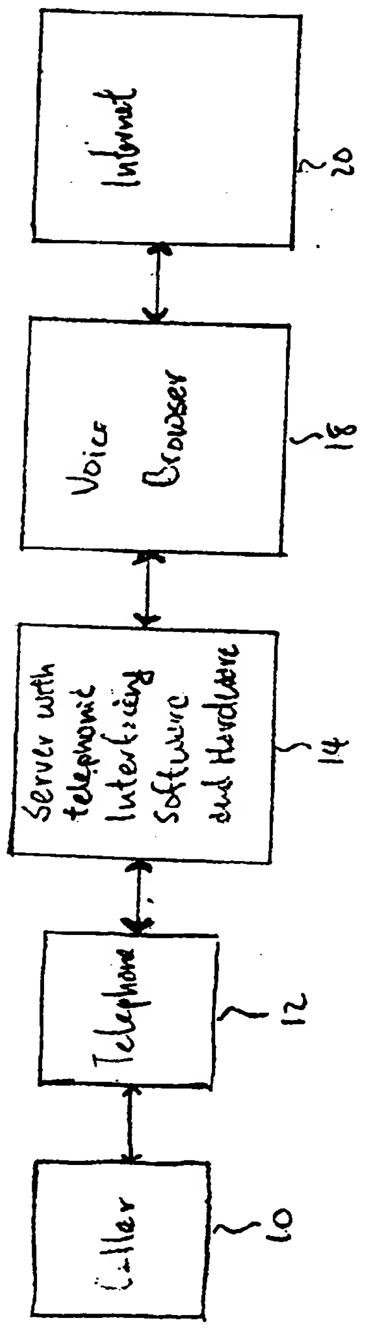

Internet
20
Voice Browser
18
Server with telephonic Interfacing Software and Hardware
14
Telephone
12
Caller
10

FIG. 1

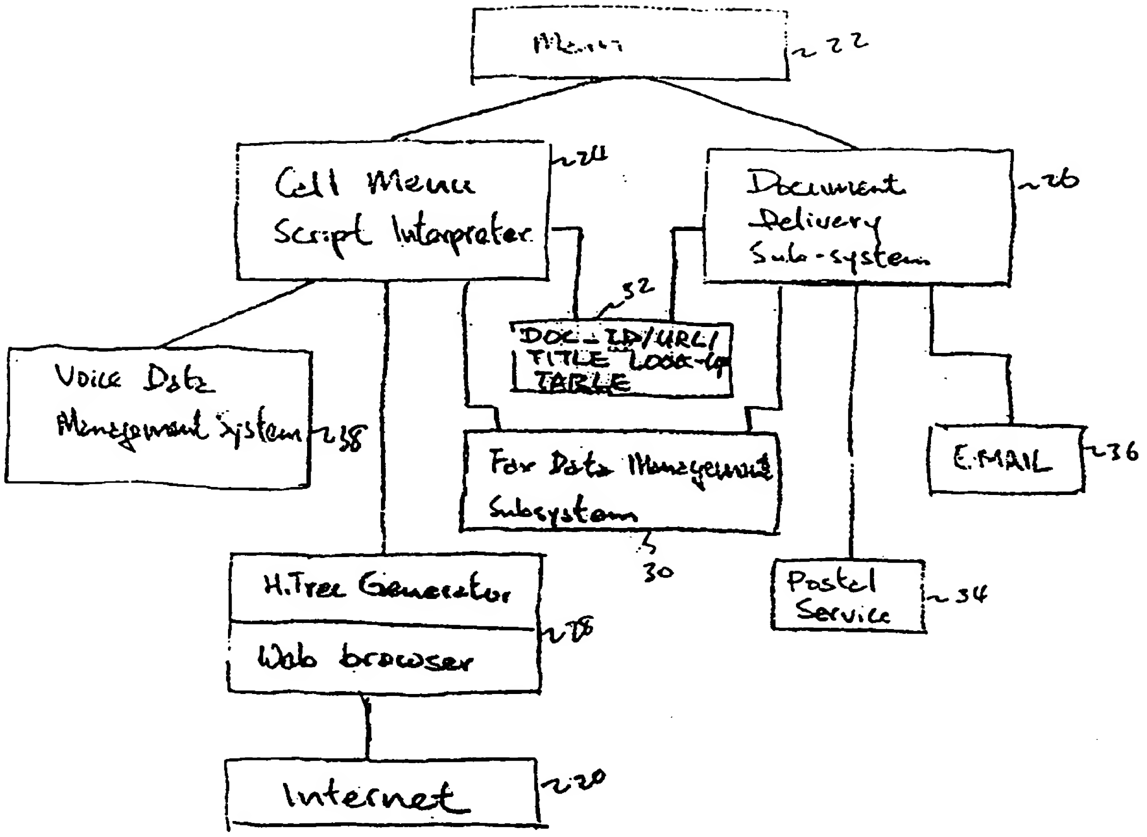

Menu
22
Cell Menu Script Interpreter
24
Document Delivery Sub-system
26
32
DOC-ID/URL/ TITLE Look-up TABLE
Voice Data Management System
38
Fax Data Management Subsystem
30
E-MAIL
36
H.Tree Generator
Web browser
28
Postal Service
34
Internet
20

Fig. 2

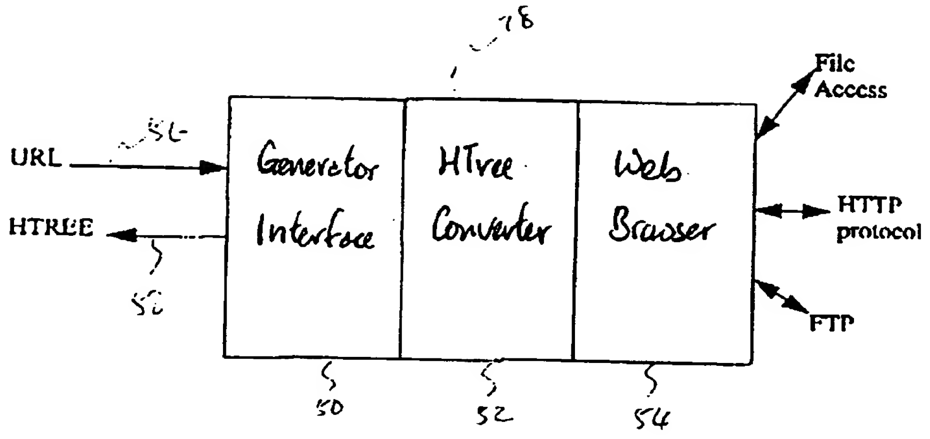

78
URL
56
HTREE
58
Generator Interface
HTree Converter
Web Browser
File Access
HTTP protocol
FTP
50
52
54

Fig. 3

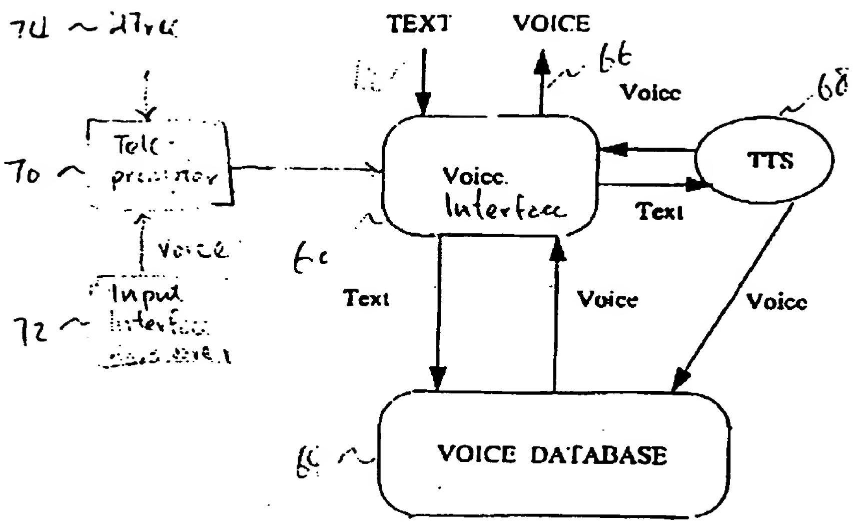

TEXT
VOICE
Voice
TTS
Voice Interface
Text
Text
Voice
Voice
VOICE DATABASE
Tele-prompter
Voice
Input Interface
74
70
72
66
68
60

Fig. 4

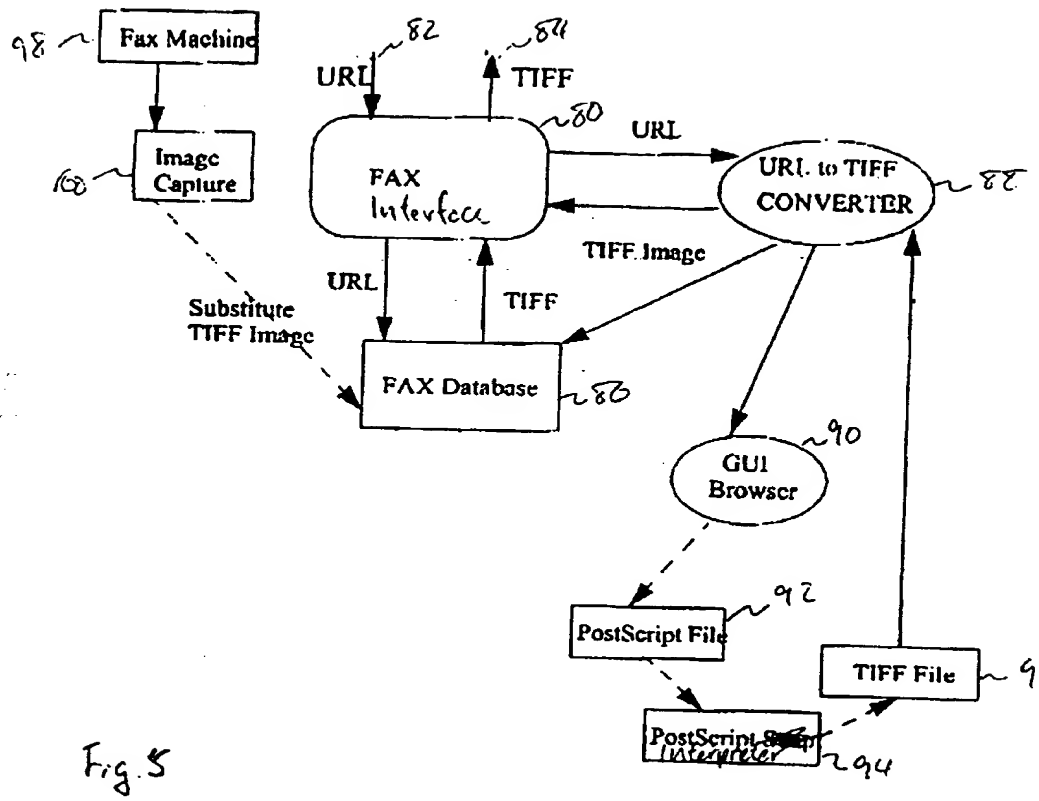

98
Fax Machine
Image Capture
100
Substitute TIFF Image
82
URL
84
TIFF
80
FAX Interface
URL
TIFF
FAX Database
86
URL
TIFF Image
URL to TIFF CONVERTER
88
GUI Browser
90
PostScript File
92
PostScript ~~Step~~ Interpreter
94
TIFF File
96

Fig. 5

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US97/03329

A. CLASSIFICATION OF SUBJECT MATTER
IPC(6) :HO4M 2/00
US CL :379/67, 88, 89, 100; 395/2.79, 2.80
According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
U.S. : 379/67, 88, 89, 100; 395/2.79, 2.80

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X --- Y	US 4,716,583 (GRONER et al.) 29 December 1987. Entire document	1, 4, 8, 20, 23, 27 ------------ 2, 3, 5-7, 9-19, 21, 22, 24-26, 28-38
Y	Makoto, "TNG/PhoneShell. (Part 2) A proposal and an implimentation of internet Access Method with Telephones and Facsimilies", JICST 96A0053311, May 1995	2, 3, 5, 6, 9, 10, 12-19, 21, 22, 24, 25, 28, 29, 31-38
Y	US 5,265,033 (VAJK et al.) 23 November 1993, col 7 lines 3-20.	7, 11, 26, 30

[✓ X] Further documents are listed in the continuation of Box C. ☐ See patent family annex.

*	Special categories of cited documents:	"T"	later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention
"A"	document defining the general state of the art which is not considered to be part of particular relevance		
"E"	earlier document published on or after the international filing date	"X"	document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone
"L"	document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)	"Y"	document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art
"O"	document referring to an oral disclosure, use, exhibition or other means		
"P"	document published prior to the international filing date but later than the priority date claimed	"&"	document member of the same patent family

Date of the actual completion of the international search 02 MAY 1997	Date of mailing of the international search report 03 JUN 1997
Name and mailing address of the ISA/US Commissioner of Patents and Trademarks Box PCT Washington, D.C. 20231 Facsimile No. (703) 305-3230	Authorized officer [illegible] DANIEL S. HUNTER Telephone No. (703) 308-6732

Form PCT/ISA/210 (second sheet)(July 1992)*

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US97/03329

C (Continuation). DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	Groner, "The Telephone-the Ultimate Terminal", Telphony, pp34-40, June 1984.	1-38
A	Arita et al., "The Voice Browser - an Archetype-Based Dialog Model", NEC Res & Develop., Vol 36 No 4. October 1995 pp 554-561	1-38
A	Hemphill et al., "Surfing the Web by Voice" ACM 0-89791-751-0-95/11, pp 215-222, November 1995.	1-38
A	Christodoulakis et al. "The Multimedia Object Presentation Manager of MINOS: A Symmetric Approach", SIGMOD Vol 15 No 2 pp295-310, June 1986	1-38
A	Zue, "Navigating the Information Superhighway Using Spoken Language Interfaces", IEEE Expert, October 1995, pp39-43.	1-38

Form PCT/ISA/210 (continuation of second sheet)(July 1992)★